2Р

УДК 004.4:519.85

К.А. Ручкин, Л.В. Холодов
Государственный университет информатики и искусственного интеллекта, Украина

Распределённая разработка программного обеспечения системы компьютерного моделирования задач хаотической динамики

В настоящее время распределённая разработка программного обеспечения приобретает всё большее значение. Это в первую очередь связано с ростом сложности создаваемых программных систем, повышением требований к функциональности и качеству конечного программного продукта, сложностью предметной области. Такие характеристики присущи системам компьютерного моделирования задач хаотической динамики. В связи с этим представляется актуальной распределённая разработка системы компьютерного моделирования и анализа решений задач хаотической динамики. В данной статье обосновывается выбор методологии распределённой разработки вышеозначенной системы.

Целью статьи является разработка системы визуализации движения твёрдого тела и исследования характера его движения. Кроме того, система должна решать задачи исследования систем дифференциальных уравнений методами, аналогичными методам исследования движения твердого тела, интерактивного и пакетного расчёта решений поставленной задачи и её характеристик, ведения пополняемой базы данных решений задач, расширения системы посредством системы подключаемых модулей – плагинов.

Исходя из собранных требований, были определены области изменчивости и расширения приложения. В качестве базовой была выбрана открытая компонентно-ориентированная архитектура, расширяемая посредством динамически подключаемых библиотек. Географическая разрозненность разработчиков определила дополнительное условие распределённости разработки, поставив тем самым задачу выбора методологии разработки.

Для организации процесса разработки были рассмотрены два класса методологий: формальные и гибкие. Учитывая высокую вариативность и расширяемость системы, а также недостаточную определённость требований к отдельным из её подсистем, были выбраны гибкие методологии разработки.

Гибкие методологии акцентируют внимание на предпочтении быстрой реакции на изменения требований следованию плану, планированию коротких этапов (1-2 месяца) и итераций (1-2 недели), тесном сотрудничестве заказчиков и разработчиков, предпочтению функционирующего (пусть даже в ограниченном объёме) программного обеспечения подробным спецификациям. К таким методологиям в первую очередь относятся SCRUM и экстремальное программирование.

Методология SCRUM на сегодняшний день является одной из самых популярных методологий разработки. Она вводит в процесс разработки три роли: scrum master, product owner, team. Scrum master является связующим звеном между менеджментом и командой. Как правило, эту роль в проекте играет менеджер проекта или team-lead. К основным его обязанностям относят: создание атмосферы доверия, устранение препятствий, оповещение команды об открытых вопросах и

проблемах, контроль за соблюдением практик и процесса в команде. Product owner – это человек, отвечающий за разработку продукта. Он является единой точкой принятия окончательных решений для команды в проекте, именно поэтому это всегда один человек, а не группа или комитет. В его обязанности входит: формирование видения проекта, формирование product backlog, расстановка приоритетов задач, предоставление требований, понятных команде, взаимодействие с командой и заказчиком, приёмка кода в конце каждой итерации. В методологии SCRUM команда является самоорганизующейся и самоуправляемой. Команда берёт на себя обязательства по выполнению работ на спринт перед product owner. Работа команды оценивается как работа единой группы. В SCRUM вклад отдельных членов команды не оценивается, так как это разрушает самоорганизацию команды. Размер команды варьируется от 5 до 9 человек. SCRUM использует следующие виды проектных документов: product backlog, sprint backlog. Product backlog – это список имеющихся на данный момент требований к разрабатываемой системе с расставленными приоритетами. Sprint backlog содержит функциональность, выбранную product owner из product backlog. Все функции разбиваются по задачам, каждая из которых оценивается командой и переносится на итерации. Результатом спринта является готовый продукт, который можно передавать заказчику. Длительность спринта – 30 дней, это позволяет обеспечивать быструю обратную связь от заказчика к команде [1].

Помимо методологии SCRUM была рассмотрена методология экстремального программирования. Главный акцент в этой методологии ставится на разработку кода как главное занятие программистов. Экстремальное программирование требует постоянного использования следующих методик: пересмотр кода (программирование парами), автоматическое тестирование кода (тестирование модулей и функциональное тестирование), рефакторинг, простота проектных решений, коллективное владение кодом, автоматическая сборка и интеграционное тестирование, короткие итерации [2]. Полный цикл разработки кода в рамках экстремального программирования состоит из: парного решения задачи, разработки кода на основе предварительно разработанных тестов, достижения работы тестовых случаев, интеграции только что написанного кода в систему и интеграционное тестирование. К сожалению, отдельные из вышеперечисленных методик не могут быть применены в условиях удалённой разработки, поэтому полноценное использование методологии экстремального программирования для данного проекта было невозможно.

Следование принципам, пропагандируемым этими методиками, позволяет создать баланс между формальными методами и свободой разработки. Однако наиболее эффективно они работают при использовании соответствующих инструментов разработки: системы контроля версий, системы отслеживания изменений в проекте, системы управления заданиями и регистрации ошибок, базы знаний проекта, системы сборки. Использование таких инструментов упрощает процесс разработки и позволяет автоматизировать рутинные операции. Следует отметить, что использование большинства из вышеперечисленных инструментов требует наличия сервера проекта, на котором они бы могли быть установлены. Если проект предполагает открытую разработку, можно воспользоваться услугами Source Forge, который бесплатно предоставляет системы Wiki, Subversion и Trac. Однако если открытая разработка невозможна либо требуется использование каких-либо дополнительных сервисов, единственным решением является установка сервера, находящегося в режиме on-line и доступного из сети Internet, что и было сделано для данного проекта.

Распределённая разработка программных проектов предполагает коллективное владение кодом. Это означает следующее: любой разработчик может вносить изменения в любой участок кода. Это требует разработки и использования единых стандартов кодирования. В зависимости от квалификации команды и выбранной культуры разработки стандарты могут включать следующие разделы: архитектурные требования, требования к физической организации проекта, требования к проектированию, требования к написанию кода на определённом языке (например, C++), требования к форматированию кода (как правило, должны быть очень лояльными), конвенции именования. Для контроля соответствия кода стандартам следует прибегать к ревизиям кода. Более продуктивными являются формальные инспекции кода, при которых код анализируется группой разработчиков, каждый из которых выполняет свою определённую роль [3].

Для снижения рисков и уточнения требований к проекту эффективным является применение на ранних этапах разработки построения прототипов решений. Проблемы принятия решений по поводу работы системы в целом, как правило, связаны с нехваткой информации о тех или иных её аспектах. Однако сбор такой информации невозможен без разработки системы, что приводит к появлению замкнутого круга. Выходом из такой ситуации является прототипирование отдельных компонентов и реализация частей системы «на выброс» [4], [5]. Кроме того, прототипы помогают новым разработчикам понять причины принятия решений, поскольку моделируют лишь часть функциональности разрабатываемой системы и позволяют рассмотреть её отдельные аспекты.

Для разрабатываемой системы в качестве базовой была выбрана методика SCRUM. Для автоматизации процесса использованы следующие инструменты: система контроля версий – Subversion [6], интегрированная web-система поддержки разработки – Edgewall Trac [7], система сборки проекта – GNU Make [8]. В качестве основного языка программирования был выбран язык C++, для которого были разработаны требования по написанию кода (стандарты кодирования) [9]. Для подсистем с высокой вариативностью начата разработка прототипов.

Литература

1. Ручкин К.А. Методы компьютерного моделирования и анализа решений задач хаотической динамики // Искусственный интеллект. – 2004. – № 4. – С. 175-181.
2. Уразбаев Асхат. Обзор методологии SCRUM. – Режим доступа: http://citforum.ru/SE/project/scrum.
3. Бек Кент. Экстремальное программирование. – СПб.: Питер, 2002. – 224 с.
4. Макконелл С. Совершенный код. Мастер-класс: Пер. с англ. – М.: Издательско-торговый дом «Русская редакция». – СПб.: Питер, 2005. – 896 с.
5. Брукс Фредерик. Мифический человекомесяц, или Как создаются программные системы: Пер. с англ. – СПб.: Символ-Плюс, 2001. – 304 с.
6. Хант Эндрю, Томас Дэвид. Программист-прагматик. Путь от подмастерья к мастеру. – М.: Издательство «Лори», 2004. – 270 с.
7. Система контроля версий Subversion. – Режим доступа: http://subversion.tigris.org.
8. Web-система поддержки разработки Edgewall Trac. – Режим доступа: http://trac.edgewall.org.
9. Система сборки проектов GNU Make. – Режим доступа: http://www.gnu.org/software/make.
10. Стандарты написания кода на C++ для разрабатываемой системы. – Режим доступа: https://unt.game-host.org/twiki/bin/view.cgi/Main/CppCodingStandarts.

Статья поступила в редакцию 30.03.2008.